(19)日本国特許庁（ＪＰ）　(12)**公開特許公報（Ａ）**　(11)特許出願公開番号

**特開平9－34823**

(43)公開日　平成９年(1997)２月７日

| (51)Int.Cl.[6] | | 識別記号 | 庁内整理番号 | ＦＩ | | 技術表示箇所 |
|---|---|---|---|---|---|---|
| Ｇ０６Ｆ | 13/00 | ３５７ | 9460－5E | Ｇ０６Ｆ | 13/00 | ３５７Ｚ |
| | | ３５５ | 9460－5E | | | ３５５ |
| | 9/44 | ５３０ | 9189－5B | | 9/44 | ５３０Ｍ |
| | 15/16 | ３７０ | | | 15/16 | ３７０Ｎ |

審査請求　未請求　請求項の数28　ＯＬ　（全 24 頁）

(21)出願番号　特願平8－65035

(22)出願日　平成８年(1996)３月21日

(31)優先権主張番号　０８／４０８６３４

(32)優先日　1995年３月22日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　595034134
サン・マイクロシステムズ・インコーポレイテッド
Ｓｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ，　Ｉｎｃ．
アメリカ合衆国カリフォルニア州94043－1100・マウンテンビュー・ガルシアアベニュー　2550

(72)発明者　ブルース　イー．　マーティン
アメリカ合衆国，　カリフォルニア州　94005，　ブリスベン，　ハンボールド　ロード　820

(74)代理人　弁理士　長谷川　芳樹　（外４名）

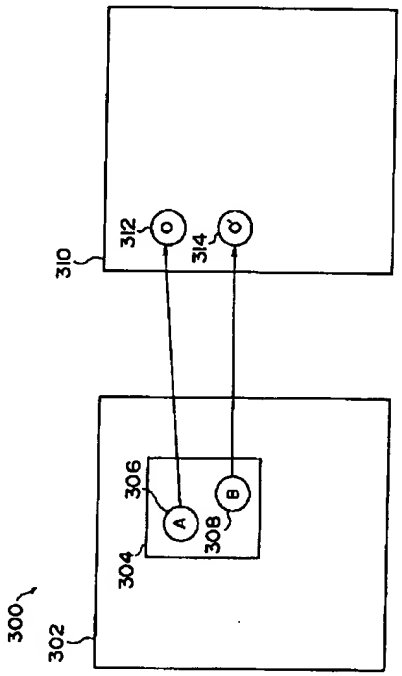
最終頁に続く

(54)【発明の名称】　分散オブジェクト環境におけるオブジェクト間の関連を管理する方法および装置

(57)【要約】

【課題】　２つ以上のオブジェクトレファレンスが同一のオブジェクトを言及しているか否かを判定する方法と装置、オブジェクトにユニーク識別子を供給するための方法と装置、関連性形成のために役割タイプをチェックする方法と装置、および役割と役割におけるオブジェクト位置を１つの関連性において保存する方法と装置を提供する。

【解決手段】　２つ以上のオブジェクトレファレンスが同一のオブジェクトを言及しているか否かの判定にあたって、ユニークオブジェクト識別子が比較され、オブジェクトレファレンスによって言及されたオブジェクトが同一であるか否かが判定される。ユニーク識別子は、プロセスＩＤ、形成時間、およびプロセスカウンターに加えて、オブジェクトを形成したプロセスのマシンアドレスを同定する情報を連鎖させることによって供給される。



1

【特許請求の範囲】

【請求項1】　第1のオブジェクトレファレンスがいずれかのコンピュータのメモリ内に格納され、前記第1のオブジェクトレファレンがいずれかのコンピュータのメモリ内で実行されるコンピュータ制御プロセス内に存在することを言及するとともに、第2のオブジェクトレファレンスがいずれかのコンピュータのメモリ内に格納され、前記第2のオブジェクトレファレンスがコンピュータメモリ内で実行されるコンピュータ制御プロセス内に存在することを言及するコンピュータネットワーク上で相互接続された複数のコンピュータを含む分散オブジェクトシステムにおいて、前記第1のオブジェクトが前記第2のオブジェクトと同一であるか否かを判定するコンピュータに実装される方法であって、

a）コンピュータ制御のもとで、前記分散オブジェクトシステムにおける前記第1のオブジェクトと前記第2のオブジェクトとを、唯一性をもって有効に識別する識別子を前記第1のオブジェクトと前記第2のオブジェクトとの各々に供給するステップと、

b）コンピュータ制御のもとで、前記第2のオブジェクトレファレンスを前記第1のオブジェクトへ転送するステップと、

c）コンピュータ制御のもとで、前記第2のオブジェクトレファレンスを用い、前記第1のオブジェクトから前記第2のオブジェクトへ前記第2のオブジェクトのユニークな識別子に関するリクエストを開始するステップと、

d）コンピュータ制御のもとで、前記第1のオブジェクトのユニークな識別子が前記第2のオブジェクトのユニークな識別子と同一であるか否かを判定するステップとを備える方法。

【請求項2】　コンピュータ制御のもとで、前記識別子が、

a）ネットワークアドレス装置と、

b）プロセス識別子装置と、

c）時間値装置と、

d）インクリメント値装置と、

の値を決定する請求項1記載の方法。

【請求項3】　前記ネットワークアドレス装置の値は、前記ネットワーク上のコンピュータのインターネットプロトコルアドレスである請求項2記載の方法。

【請求項4】　各オブジェクトに関して識別子を提供する前記ステップは、

a）コンピュータ制御のもとで、前記オブジェクトを内部で形成するコンピュータに関する前記インターネットプロトコルアドレスを決定するステップと、

b）コンピュータ制御のもとで、前記オブジェクトを形成するプロセスの前記プロセス識別子装置の値を決定するステップと、

c）コンピュータ制御のもとで、現在のローカルシステム時間を決定するステップと、

2

d）コンピュータ制御のもとで、前記インクリメント値装置の現在値を決定するステップと、

e）前記識別子を形成するため、コンピュータ制御のもとで、前記インターネットプロトコルアドレス、前記プロセス識別子装置値、前記現在のローカルシステム時間、および前記現在値を連鎖させるステップとを備える請求項3記載の方法。

【請求項5】　コンピュータ制御のもとで、前記オブジェクトに関連するメモリ空間内に前記識別子を格納するステップを更に備え、前記メモリ空間は前記ネットワーク上のいずれかのコンピュータに含まれる請求項4記載の方法。

【請求項6】　前記オブジェクトとともに格納されている常駐データ構造の中に前記識別子が格納される請求項5記載の方法。

【請求項7】　分散オブジェクトシステムにおいて、ネットワーク上で相互接続された1台以上のコンピュータ上でコンピュータ制御のもとで実行されている1つ以上のプロセス内に存在する2つ以上のオブジェクト間に形成するべき所望の関連性の有効性をチェックする、コンピュータに実装される方法であって、

a）コンピュータ制御のもとで、関連させるべき各オブジェクトから、前記オブジェクト間に前記所望の関連性を形成するのに有効な関連性ファクトリ機構へ役割を転送するステップと、

b）コンピュータ制御のもとで、前記関連性ファクトリ機構へ転送された各役割が前記所望の関連性にとって適切なタイプのものであるか否かを判定するステップとを備える方法。

【請求項8】　コンピュータ制御のもとで、前記関連性ファクトリ機構へ転送された各役割が前記所望の関連性にとって適切なタイプのものであるか否かを判定する前記ステップは、コンピュータに実装された、

a）コンピュータ制御のもとで、タイプチェックアレイと期待アレイとを形成するステップと、

b）コンピュータ制御のもとで、前記関連性ファクトリ機構へ転送された各役割に関する、下記のステップを含むタイプチェックをコンピュータ制御のもとで行うステップであって、

i）コンピュータ制御のもとで、前記関連性ファクトリ機構へ転送された1つの役割を、期待される役割タイプの少なくとも1つと比較し、コンピュータ制御のもとで、前記関連性ファクトリ機構へ転送された前記役割が、期待される前記役割タイプと同じタイプであるか否かを判定するステップと、

ii）前記役割が期待される前記役割タイプと同じタイプであるとのコンピュータ制御のもとでの判定に応じて、コンピュータ制御のもとで、前記タイプチェックアレイのひとつの要素をインクリメントさせるステップ

と、を備えるステップと、
ｃ）前記関連性ファクトリ機構へ転送された前記役割タイプが前記関連性ファクトリ機構によって期待されるタイプに一致するか否かを判定するために、コンピュータ制御のもとで、前記タイプチェックアレイを前記期待アレイと比較するステップとを備える請求項７記載の方法。
【請求項９】　コンピュータ制御のもとで、前記比較するステップが、コンピュータ制御のもとで前記関連性ファクトリ機構へ転送された前記役割を、前記分散オブジェクトシステム中に存在するインターフェイス保存機構から検索するステップを備える請求項８記載の方法。
【請求項１０】　コンピュータ制御のもとで、前記比較するステップが、前記役割に関連するオブジェクトに問い合せて、前記オブジェクトに関するインターフェイスを検索するステップを備える請求項９記載の方法。
【請求項１１】　第１のオブジェクト役割を有する第１のオブジェクトに関するオブジェクトレファレンスを得る、コンピュータに実装された方法であって、
ａ）コンピュータ制御のもとで、第２のオブジェクト役割へ関連性オブジェクトの名称と前記第１のオブジェクト役割の役割名とを転送するステップと、
ｂ）前記第１のオブジェクトに関する前記オブジェクトレファレンスを決定するため、コンピュータ制御のもとで、前記関連性オブジェクト名と前記役割名とを前記第２のオブジェクト役割とともに格納されたデータ構造中に格納されているエントリと比較ステップと、
を備え、前記第１のオブジェクトが前記第２のオブジェクト役割を有する第２のオブジェクトに前記関連性オブジェクトによって関連付けられる方法。
【請求項１２】　ａ）コンピュータ制御のもとで、前記関連性オブジェクトに前記第１のオブジェクト役割名を問い合せるステップと、
ｂ）コンピュータ制御のもとで、前記第１のオブジェクト役割に前記第１のオブジェクトに関する前記オブジェクトレファレンスを問い合せるステップと、
ｃ）コンピュータ制御のもとでの前記第１のオブジェクト役割名が前記データ構造中に含まれていない旨の判定に応答して、コンピュータ制御のもとで、前記オブジェクトレファレンスを前記データ構造中に格納するステップとを更に備える請求項１１記載の方法。
【請求項１３】　第１のオブジェクトに関連付けられた第１のオブジェクト役割に関するオブジェクトレファレンスを得るコンピュータに実装される方法であって、
ａ）コンピュータ制御のもとで、第２のオブジェクト役割へ前記関連性オブジェクトの名称と前記第１のオブジェクト役割の役割名とを転送するステップと、
ｂ）前記第１のオブジェクト役割に関する前記オブジェクトレファレンスを決定するため、コンピュータ制御のもとで、前記関連性オブジェクト名および前記役割名を



前記第２のオブジェクト役割とともに格納されたデータ構造の中に格納されたエントリと比較するステップとを備え、前記第１のオブジェクトを前記第２のオブジェクト役割を有する第２のオブジェクトに関連性オブジェクトによって関連付ける方法。
【請求項１４】　前記関連性に前記第１のオブジェクト役割の前記役割名を問い合せる、コンピュータに実装されるステップと、
前記役割名を、前記役割とともに格納されたデータ構造の中に格納する、コンピュータに実装されるステップとを更に備える請求項１３記載の方法。
【請求項１５】　分散オブジェクトシステムにおいて、相互接続された１台以上のコンピュータで実行されるプロセス内に存在する２つのオブジェクトレファレンスが、同一のオブジェクトを言及しているか否かを判定する装置であって、
ａ）第１のオブジェクトに関連する第１のオブジェクトレファレンス機構および第２のオブジェクトに関連する第２のオブジェクトレファレンス機構と、
ｂ）前記分散オブジェクトシステムにおいて、前記第１と第２のオブジェクトを唯一性をもって有効に識別する前記第１および第２のオブジェクトについての識別子機構と、
ｃ）コンピュータ制御のもとで、通信媒体を介して前記第２のオブジェクトレファレンスを前記第１のオブジェクトへ転送し、前記第１のオブジェクトから前記通信媒体を介して前記第２のオブジェクトへ前記第２のオブジェクトのユニーク識別子に関するリクエストを送る前記通信媒体と、
ｄ）コンピュータ制御のもとで、前記第１のオブジェクトのユニーク識別子が前記第２のオブジェクトのユニーク識別子と同一であるか否かを判定する同一性チェック機構とを備える装置。
【請求項１６】　前記識別子機構は、
ａ）ネットワークアドレス装置と、
ｂ）プロセス識別子装置と、
ｃ）時間値装置と、
ｄ）インクリメント値装置とを備える請求項１５記載の装置。
【請求項１７】　前記ネットワークアドレス装置は、インターネットアドレスで識別される請求項１６記載の装置。
【請求項１８】　前記識別子機構が、前記オブジェクトに関連するメモリ空間に配置されている請求項１６記載の装置。
【請求項１９】　前記識別子機構が、前記オブジェクトとともに格納されている常駐データ構造の中に配置されている請求項１８記載の装置。
【請求項２０】　分散オブジェクトシステムにおいて、相互接続された１台以上のコンピュータで実行されるプ

5

ロセス内に存在する２つ以上のオブジェクト間に形成されるべき所望の関連性の有効性をチェックする装置であって、

ａ）コンピュータ制御のもとで、関連付けられるべき各オブジェクトから前記所望の関連性を前記オブジェクト間に形成するのに有効な関連性ファクトリ機構へ役割を転送するための通信媒体と、

ｂ）コンピュータ制御のもとで、適正数の前記所望の関連性が前記関連性ファクトリへ転送されたか否か、または、前記関連性ファクトリへ転送された各役割が前記所望の関連性にとって適切なタイプであるか否かを判定する関連性無矛盾性評価機構とを備える装置。

【請求項２１】　前記関連性無矛盾性評価機構は、コンピュータ制御のもとで、前記関連性ファクトリ機構へ転送された各役割が前記所望の関連性にとって適切なタイプであるか否かを判定するタイプチェック機構を備え、前記タイプチェック機構は、

ａ）コンピュータ制御のもとで、タイプチェックアレイと期待アレイとを作成するアレイ作成器と、

ｂ）前記ファクトリへ転送された各役割に関するタイプチェック器であって、

ｉ）コンピュータ制御のもとで、前記関連性ファクトリ機構へ転送された役割を少なくとも１つの期待される役割タイプと比較し、前記関連性ファクトリ機構へ転送された前記役割が前記期待される役割タイプと同じタイプであるか否かを判定するための役割比較器と、

ｉｉ）コンピュータ制御のもとで、前記役割が前記期待される役割タイプと同じタイプである旨の判定に応答して前記タイプチェックアレイの１つの要素をインクリメントさせるインクリメント器と、

を備えるタイプチェック器と、

ｃ）コンピュータ制御のもとで、前記タイプチェックアレイを前記期待アレイと比較し、前記関連性ファクトリ機構へ転送された前記役割タイプが前記関連性ファクトリによって期待される役割タイプに一致するか否かをコンピュータ制御のもとで判定するアレイ比較器と、

を備える請求項２０記載の装置。

【請求項２２】　前記役割比較器は、コンピュータ制御のもとで、前記分散オブジェクトシステムにおけるいずれかのコンピュータのメモリ空間に配置されたインターフェイス格納場所から前記関連性ファクトリ機構へ転送された前記役割に関するインターフェイスを検索するインターフェイス検索機構を備える請求項２１記載の装置。

【請求項２３】　前記役割比較器は、前記役割対応するオブジェクトに前記オブジェクトに関するインターフェイスを検索するよう問い合せる問合機構を備える請求項２１記載の装置。

【請求項２４】　第１のオブジェクトは第２のオブジェクト役割を有する第２のオブジェクトに関連性オブジェクトによって関連付けられ、前記第１のオブジェクトおよび前記第２のオブジェクトは分散オブジェクトシステムにおける１台以上のコンピュータ中のコンピュータ制御プロセス中に存在する状態で、第１のオブジェクト役割を有する前記第１のオブジェクトに関するオブジェクトレファレンスを得る装置であって、

ａ）コンピュータ制御のもとで、前記関連性オブジェクトの名称と前記第１のオブジェクト役割の役割名とを前記第２のオブジェクトへ転送する通信媒体と、

ｂ）コンピュータ制御のもとで、前記関連性オブジェクト名および前記役割名と前記第２のオブジェクト役割とともに格納されているデータ構造の中に含まれているエントリとを比較し、前記第２のオブジェクトに関する前記オブジェクトレファレンスを決定する比較器と、

を備える装置。

【請求項２５】　前記第１のオブジェクト役割名が前記データ構造に含まれていない旨の判定に応答して、

ａ）コンピュータ制御のもとで、前記関連性に前記第１のオブジェクト役割名を尋ね、前記第１のオブジェクト役割に前記第１のオブジェクトに関する前記オブジェクトレファレンスを問い合せる問合機構と、

ｂ）コンピュータ制御のもとで、前記オブジェクトレファレンスを前記データ構造に格納する格納機構とを更に備える請求項２４記載の装置。

【請求項２６】　第１のオブジェクトは第２のオブジェクト役割を有する第２のオブジェクトへ関連性オブジェクトによって関連付けられており、前記第１のオブジェクトおよび前記第２のオブジェクトは分散オブジェクトシステムにおいて１台以上のコンピュータ中のコンピュータ制御プロセス中に存在する状態で、第１のオブジェクトに関連付けられた第１のオブジェクト役割に関するオブジェクトレファレンスを得る装置であって、

ａ）コンピュータ制御のもとで、前記関連性オブジェクトの名称と前記第１のオブジェクト役割の役割名とを前記第１のオブジェクト役割へ転送するための通信媒体と、

ｂ）コンピュータ制御のもとで、前記関連性オブジェクト名と前記役割名とを前記第２のオブジェクト役割とともに格納されているデータ構造の中に含まれているエントリと比較し、前記第１のオブジェクト役割に関する前記オブジェクトレファレンスを決定する比較器とを備える装置。

【請求項２７】　コンピュータ制御のもとで、前記関連性に前記第１のオブジェクト役割の前記役割名を問い合せ、前記役割とともに格納されたデータ構造に前記役割名を格納する問合機構を更に備える請求項２６記載の装置。

【請求項２８】　関連性を介して第２のオブジェクトに関連付けられているオブジェクトに関連する役割オブジェクトであって、前記第２のオブジェクトは第２の役割

7

オブジェクトに関連付けられており、前記役割オブジェクトは、前記関連オブジェクトに関するオブジェクトレファレンスと前記第2の役割オブジェクトに関するオブジェクトレファレンスとを含む格納場所を有し、前記オブジェクトは分散オブジェクトシステムにおけるネットワークにおいて相互接続された1台以上のコンピュータ上で走るひとつ以上のプロセス中に存在する役割オブジェクト。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、分散コンピューティングシステム、クライアント-サーバーコンピューティング、およびオブジェクト指向プログラミングの分野に係り、特に、分散オブジェクト環境において、関連しているオブジェクトを管理する方法および装置に関するものである。

【０００２】

【従来の技術および発明が解決しようとする課題】ここ数年来、従来のプログラミング手法で開発されたソフトウェアの納期遅れと予算超過の傾向が強まるにつれて、オブジェクト指向プログラミング手法への関心が高まっている（Taylor, D. 1990. Object Oriented Technology: A Manager's Guide. Addison Wesley. 、Gibbs, W.W. 1994. Trends in Computing: Software's Chronic Crisis. Scientific American 271(3):86-95.参照）。従来のプログラミング手法に伴う一つの問題は、多くの課題に関して、設計及び保守がしばしば著しく困難な手順モデルと「線形」コードとに重点が置かれていることに起因している。一般的に、従来の手法を用いて作成された大きなプログラムは「脆い（brittle）」。すなわち、わずかな変更でさえも、プログラミングコードの全要素が影響を受ける。従って、ユーザの要望に応じたソフトウェアの小規模な変更でも、全プログラムの大幅な再設計と書き換えとが必要になることがある。

【０００３】オブジェクト指向プログラミング戦略は、こうした問題の回避に向いている。その理由は、オブジェクト手法は、手順よりもむしろデータの取扱いに重点を置くので、プログラマに現実世界の問題をモデル化するにあたって、より直感的なアプローチを提供するからである。更に、オブジェクトは関連データと手順とをカプセル化しており、オブジェクトのインターフェイスを介してのみ関連するデータおよび手順へのアクセスを可能にすることによって、これらの情報がプログラムの他の部分から隠されている。したがって、オブジェクトのデータおよび／または手順についての変更は、プログラムの他の部分から比較的隔離された状態にある。こうして、特定のオブジェクトのコードの変更が、他のオブジェクトのコードに影響を及ぼすことがないので、従来の方法を用いて書かれたコードに比べて保守の容易なコードが提供される。更に、オブジェクトは本来モジュール

8

的な性質であり、個々のオブジェクトを種々のプログラムで再使用することができる。したがって、プログラマは「試行正（tried and true）」のオブジェクトのライブラリを作成し、種々のアプリケーションにオブジェクトを反復して使用することができる。このことにより、ソフトウェアの信頼性が高まるとともに、信頼性のあるプログラミングコードが繰り返し用いられるので、開発時間が短縮される。

【０００４】しかし、オブジェクト指向方法論の利点を十全に活用すること、特に、モジュール性によってもたらされる利点の達成は今後の課題である。特に、異なるプログラム言語によって作成されたオブジェクトや相互にネットワーク化された多数の計算プラットフォーム上にあるオブジェクトに対して、プログラマや他のユーザがユーザのプログラミングコードを大幅に変更することなく透明性のよい手法でアクセスできるようにすることが大いに望ましい。

【０００５】オブジェクト-サーバ（object-server）にリクエストを行うオブジェクト-クライアント（object-client）にオブジェクト-サーバーまたはオブジェクトがインターフェイスを提供するという、クライアント-サーバーモデルに基づくオブジェクト指向分散システムを用いて、そのような手段を提供する試みがなされている。通常、こうしたシステムにおいて、サーバはデータと関連方法から成るオブジェクトである。オブジェクト-クライアントはオブジェクト-サーバに分散システムによって媒介されるコールを実行することによって、オブジェクト-サーバの機能にアクセスすることができる。オブジェクト-サーバがコールを受けると、オブジェクト-サーバは適当な方法を実行し、結果をオブジェクト-クライアントに送り返す。オブジェクト-クライアントとオブジェクト-サーバとはオブジェクトリクエストブローカ（Object Request Broker；ＯＲＢ）を介して通信を行い、ＯＲＢは種々の分散したオブジェクトの存在場所をつきとめ、それらとの間で通信を行う（Rao, B. R. 1993. C++ and the OOP Paradigm. McGraw-Hill.参照）。

【０００６】オブジェクトのインターフェイスをオブジェクトの実装（implementation）から分離し、ソフトウェア設計者がオブジェクトの実装の詳細を気にすることなく、設計者が種々のオブジェクトの機能を利用可能とする、分散システムにおけるオブジェクトメタファ（object metaphor）は有用な技法である。プログラマはオブジェクトのインターフェイスのみを気にすればよい。更に、オブジェクト指向分散システムは、1つのインターフェイスの多重実装（multiple implementation）を可能とする。すなわち、このインターフェイスは、ネットワークを介して既に接続されている複数の異なる計算プラットフォームに配置されてもよい。したがって、ネットワークの特定のマシンで作業をしているプログラマ

は、遠隔のオブジェクトがアクセスされ、そのオブジェクトのデータが適当な時に送り返され、プログラマのコードが正しく機能するという確信を持って、プログラマが詳細な知識を持たない特定のオブジェクトをコールすることができる。こうして、システムは、モジュール化およびカプセル化の利点を十全に活用し、オブジェクト指向方法論の固有の長所を最大限に引き出すことができる。

【０００７】オブジェクト指向プログラミング技法の利点を更に拡大する努力には、プログラミングオブジェクト（programming object）のモジュール性を高める手法の開発が含まれる。そのような努力には、オブジェクト間の関連（relations）または関連性（relationships）の概念の開発が含まれる。ここで、オブジェクトの関連性とは、１つまたは複数のクラスから見たオブジェクト間の対応のことである（Rumbaugh, J., et al.. 1991. Object-Oriented Modeling and Design. Prentice Hall.参照）。関連性を用いることにより、オブジェクトプログラマは、関連性を用いなければオブジェクトの実行コードに埋もれてしまうであろうオブジェクト間の対応と制限とを明確に表現し、それによってオブジェクトのモジュール性を低減することができる。関連性を用いることによってもたらされるオブジェクト間の言及顕在化（reference externalization）は、オブジェクト指向コンピューティングモデルをデーターベース理論に見られるエンティティ関連モデルに結び付ける、対称的かつ非冗長なモデル化を実現可能とする。システムを非常に自然な用語で考えることができるので、こうした戦略は関連オブジェクトのシステムのモデル化に特に有用である。例えば、多くのコンピュータシステムにおけるファイル格納によく用いられるメタファ（metaphor）である特定のフォルダに含まれているドキュメントのモデル化は、「包含（containment）」関係によって、フォルダーオブジェクト（folder object）をドキュメントオブジェクト（document object）に関連付けることによってモデル化できる。こうしたモデルは、ドキュメントが格納されるフォルダの現実世界の場合に対応して非常に自然な手法で関連性を表現する。

【０００８】これらの概念を分散オブジェクト指向システムに実装することは、依然として困難な仕事である。例えば、複数のオブジェクトレファレンスが同じオブジェクトに向けられているか否かを決定する方法が必要になる。こうした決定を行うことの必要性は、オブジェクト間の関連性の実装実行にも関連している。通常、関連オブジェクト間の関連の連鎖を「マップ（map）」することは、プログラマまたはシステム管理者にとって有用である。

【０００９】更に、関連付けの手法と分散オブジェクト指向システムとの実現に関しては、論理的に首尾一貫したやり方でオブジェクト間に関連性が確実に構成されるような方法が存在しなければならない。換言すれば、例えば、ドキュメント自体がフォルダに対する包含関係の中に配置されることを防止するような方法が強く望まれる。最後に、オペレーティングシステムによって課せられるオーバーヘッドを削減するような効率の高いやり方で関連性のあるオブジェクトを識別する方法を提供することも望ましい。

【００１０】

【課題を解決するための手段】本発明は、分散オブジェクトシステムにおけるオブジェクト間の関連を管理するための方法および装置に関する。本発明の方法および装置によれば、同一のオブジェクトに言及するオブジェクトレファレンスを識別する必要性からシステムに課せられる制限が、論理的に無矛盾なオブジェクト間の関連性の構築を提供するとともに、関連性を介して対応付けられたオブジェクトの効率的な識別を提供することによりシステム効率を高める。

【００１１】本発明は、コンピュータのメモリで実行されるプロセス内に置かれる第１のオブジェクトレファレンスを含む分散オブジェクトシステムにおいて、第１のオブジェクトが第２のオブジェクトと同一であるか否かを判定する、コンピュータに実装される方法を含む。ここで、第１のオブジェクトレファレンスは第１のオブジェクトに言及し、いずれかのコンピュータのメモリに格納された第２のオブジェクトレファレンスは第２のオブジェクトに言及する。この方法は、まず、分散オブジェクトシステム内におけるオブジェクトをユニークに識別するのに有効な、第１オブジェクトと第２のオブジェクトとに関する夫々の識別子を提供する。次に、第２のオブジェクトに対するオブジェクトレファレンスが第１のオブジェクトに転送され、第１のオブジェクトが第２のオブジェクトに対して、第２のオブジェクトのユニークな識別子に関するリクエストを開始する。次いで、第１のオブジェクトは、転送された識別子が第２のオブジェクトのユニークな識別子に一致するかどうかを判断する。ユニークな識別子は、ネットワークアドレス、プロセス識別子、時間値、及びインクレメンター値を有することが好適である。

【００１２】本発明は、分散オブジェクトシステムにおいて、２複数のオブジェクト間に構成すべき関連性の無矛盾性をチェックする、コンピュータで実装実行される方法を提供する。本発明の方法では、コンピュータ制御下の役割を、関連させるべき各オブジェクトからオブジェクト間に有効な関連性を構成する関連性ファクトリ機構へ転送する。関連性ファクトリへ転送された各役割が、必要な関連性のために適当なタイプのものであるか否かに関する判定が行われる。本発明の方法は、タイプチェックアレイの作成、期待アレイの作成、およびファクトリに転送された各役割のタイプチェックの実行という各ステップを備えることが好ましい。タイプチェック

は、関連性ファクトリへ転送された役割を、期待される役割タイプの少なくとも1つと比較することにより、転送された役割が、期待される役割タイプと同じタイプであるか否かを判定し、転送された役割が期待される役割タイプと同じタイプであるとの判定に応答して、タイプチェックアレイの1つの要素にインクリメントを加える。次に、タイプチェックアレイを期待アレイと比較して、関連性ファクトリへ転送された役割タイプが関連性ファクトリの期待する役割タイプに一致するか否かを判定する。

【0013】本発明は、第1のオブジェクト役割を有する第1のオブジェクトに関するオブジェクトレファレンスを得るためコンピュータで実装実行される方法を含む。ここで、第1オブジェクトは第2のオブジェクト役割を有する第2オブジェクトに関連し、第1と第2のオブジェクトは関連性オブジェクトによって関連付けられている。この方法においては、関連性オブジェクトのオブジェクトレファレンスと第1のオブジェクト役割の役割名は、第2のオブジェクト役割へ転送される。関連性オブジェクトレファレンスと第1のオブジェクト役割名は、第2のオブジェクト役割に保存されているデータ構造内に含まれているエントリーと比較され、データ構造が第1のオブジェクトに関するオブジェクトレファレンスを含むか否かを判定する。この方法では、更に、第1のオブジェクト役割名が第2のオブジェクト役割に保存されているデータ構造中に含まれていないとの判定に応答して第1のオブジェクトに関するオブジェクトレファレンスのみならず関連性オブジェクトの第1のオブジェクト役割名を関連性オブジェクトに問い合わせることが好適である。次に、第1のオブジェクトに関するオブジェクトレファレンスは、第2のオブジェクト役割に格納されているデータ構造中に格納される。

【0014】本発明は、関連性オブジェクトによって第2のオブジェクト役割を有する第2のオブジェクトに関連づけられている第1のオブジェクトに関連付けられている第1のオブジェクト役割に関するオブジェクトレファレンスを得る、コンピュータで実装実行される方法を含む。この方法において、関連性オブジェクトのオブジェクトレファレンスと第1のオブジェクト役割の役割名は第2のオブジェクト役割へ転送される。これらは、第2のオブジェクト役割に格納されているデータ構造中に含まれているエントリと比較され、第1のオブジェクト役割に関するオブジェクトレファレンスがデータ構造に含まれているか否かを判定する。

【0015】また、本発明は、分散オブジェクトシステム中の2つのオブジェクトレファレンスが同一のオブジェクトを言及しているか否かを判定する装置を提供する。この装置は、第1のオブジェクトに関連する第1のオブジェクトレファレンスと、第2のオブジェクトに関連する第2のオブジェクトレファレンスと、第1及び第



2のオブジェクトの各々のための識別子（分散オブジェクトシステムにおける各オブジェクトをユニークに識別するのに有効である）を提供する識別機構（identity mechanism）と、第2のオブジェクトレファレンスを第1のオブジェクトへの転送および第2のオブジェクトのユニーク識別子に関する第1のオブジェクトから第2のオブジェクトへのリクエストを開始する通信媒体と、第1のオブジェクトのユニーク識別子が第2のオブジェクトのユニーク識別子と同一であるか否かを判定するための同一性チェック機構とを備える。

【0016】また、本発明は、複数のオブジェクトおよび分散オブジェクトシステムの相互間で構成されるべき所要の関連性の有効性をチェックする装置を含む。この装置は、関連付けられるべき各オブジェクトからの役割を、オブジェクト間に所望の関連性を構成するのに有効な関連性ファクトリ機構へ転送するための通信媒体を備える。無矛盾評価機構が、所望の関連性のための妥当な数の役割が関連性ファクトリ機構へ転送されたか否か、また、関連性ファクトリ機構へ転送された各役割が所望の関連性にとって妥当なタイプであるか否かを判定する。

【0017】また、本発明は、第1のオブジェクト役割を有する第1のオブジェクトに関するオブジェクトレファレンスを得る装置を含む。ここで、第1のオブジェクトは関連性オブジェクトによって第2のオブジェクト役割を有する第2のオブジェクトに関連付けられている。この装置は、関連性オブジェクトのオブジェクトレファレンスと第1のオブジェクト役割の役割名を第2のオブジェクト役割へ転送する通信媒体と、関連性オブジェクトレファレンスと第1のオブジェクト役割名とを第2のオブジェクト役割に格納されているデータ構造の中に含まれているエントリと比較し、第1のオブジェクト役割に関するオブジェクトレファレンスがデータ構造に含まれているか否かを判定する比較器とを備える。

【0018】更に、本発明は、関連性オブジェクトによって第2のオブジェクト役割を有する第2のオブジェクトに関連付けられている第1のオブジェクトに関するオブジェクトレファレンスを得るための装置を含む。この装置は、関連性オブジェクトのオブジェクトレファレンスと第1のオブジェクト役割の役割名を第2のオブジェクト役割へ転送する通信媒体と、関連性オブジェクトレファレンスと第1のオブジェクト役割名とを第2のオブジェクト役割に格納保存されているデータ構造内に含まれているエントリと比較して第1のオブジェクト役割に関するオブジェクトレファレンスがデータ構造内に含まれているか否かを判定するための比較器とを備える。

【0019】

【発明の実施の形態】本発明は、分散オブジェクト環境において分散しているオブジェクト間の関連性を管理するための方法と装置とを含んでいる。本発明の方法と装

置を用いて、分散オブジェクト環境において分散しているオブジェクトを、様々なプラットフォーム、オペレーティングシステム、およびマシンについて、無矛盾な効率的な方法で処理することができる。

【００２０】以下、添付図面を参照しながら、本発明の実施形態を説明する。なお、図面の説明にあたって同一の要素には同一の符号を付し、重複する説明を省略する。

【００２１】Ｉ．用語の定義

本明細書において使用する用語の定義について説明する。

【００２２】「分散オブジェクト」または「オブジェクト」という用語は、インターフェイス分散オブジェクト（interface distributed object）で定義される作用によって操作することのできるカプセル化されたコードとデータとのパッケージをいう。したがって、当業者にとっては、分散オブジェクトは、従来のプログラミングオブジェクトを定義する基本的特性を含むもの見えるあろう。しかし、分散オブジェクトは、以下の２つの重要な特徴を含むことをもって、従来のプログラミングオブジェクトとは異なる。第１の特徴は、分散オブジェクトが多言語的（multilingual）であることである。分散オブジェクトのインターフェイスは、様々なプログラミング言語にマップされ得るインターフェイス定義言語を用いて定義される。こうしたインターフェイス定義言語のひとつがＩＤＬである。第２の特徴は、分散オブジェクトは場所独立性がある、すなわち、分散オブジェクトはネットワーク内のどこにでも配置され得ることである。このことは、単一のアドレス空間内、例えばクライアントのアドレス空間内に存在する従来のプログラミングオブジェクトに対して著しく対照的である。分散オブジェクトは、リクエストを他のオブジェクトへ送っているか、他のオブジェクトからのリクエストに応答しているかによって、オブジェクト-クライアントかオブジェクト-サーバーかの何れかとなる。リクエストおよび応答は、オブジェクトの場所と状態とを認識しているＯＲＢを介して行われる。

【００２３】「分散オブジェクトシステム」または「分散オブジェクト環境」は、分散オブジェクトシステム内のオブジェクトの場所と状態を認識しているＯＲＢを介して通信を行う複数の遠隔オブジェクトから成るシステムをいう。好ましい実施例においては、オブジェクトは、本出願人の別出願である、「弁理士整理番号：Ｐ７１７／ＳＵＮ０２３、発明の名称：オブジェクトの集合の管理方法及び装置、発明者：Dwight F. Hare 他、１９９５年３月２２日　米国出願」；「弁理士整理番号：Ｐ７１５／ＳＵＮ０１８、発明の名称：分散オブジェクトシステムにおけるオブジェクト間の通信の接続を管理する方法及び装置、発明者：David Brownell他、１９９５年３月２２日　米国出願」、および、「弁理士整理番号：Ｐ７４７／ＳＵＮ０３０、発明の名称：コンピュータプロセスを管理する方法および装置、発明者：Anthony W. Menges 他、１９９５年３月２２日　米国出願」に記載方法を用いて通信を行う。こうしたＯＲＢを実行するための好ましいシステムアーキテクチャは、共通のオブジェクトリクエストブローカーアーキテクチャ（Common Object Request Broker Architecture；ＣＯＲＢＡ）仕様によって提供される。ＣＯＲＢＡ仕様は、サンマイクロシステムズ社、ディジタルエレクトロニクス社、ハイパーデスク社、ヒューレットパッカード社、サンソフト社、ＮＣＲ社、オブジェクトデザイン社を含むベンダの協会であるオブジェクトマネージメントグループ（Object management Group；ＯＭＧ）定義され、こうしたサービスの提供を要求するクライアントに提供されている。

【００２４】「オブジェクトレファレンス」または「オブジェレフ（objref，以下、「オブジェクトレファレンス」とする。）」は、他のオブジェクトへのポインタを含む。オブジェクトレファレンスの形成と定義は当業者には周知である（InterfaceUser's Guide and Reference: Project DOE External Developer's Release 2.1994. SunSoft.参照）。

【００２５】「クライアント」は他のオブジェクトへリクエストを送る実体であり、この他のオブジェクトは「サーバ」と呼ばれ、クライアントはサーバに作用（operations）または実装実行（implementations）を発動させる。分散オブジェクト環境においては、オブジェクトに対する多言語性の要請から、クライアントは実装プログラミング言語の知識を持つ必要がなく、実装オブジェクトはクライアントのプログラミング言語の知識を持つ必要もない。分散オブジェクト環境におけるクライアントとサーバは、インターフェイス定義言語のみを用いて通信すればよい。上述のように、クライアントによるサーバーへのリクエストとクライアントへのサーバーの応答は、ＯＲＢによって処理される。

【００２６】「オブジェクトインターフェイス」は、あるオブジェクトが提供する作用、属性、および例外の仕様である。分散オブジェクトのオブジェクトインターフェイスは、例えばＩＤＬを用いて書くことが望ましい。上述のように、オブジェクトは、それらのインターフェイスを介したトランザクションを実行する。したがって、インターフェイスの使用は、トランザクション中のオブジェクトの方法とオデータとを定義するプログラミング言語を考慮するというオブジェクトへ要請を軽減する。

【００２７】「関連性」または「関連性オブジェクト」は、関連するオブジェクトの実装実行コードで別途に記述される複数のオブジェクト間に対応関係を形成するオブジェクトである（Rumbaugh, J. 1987. Relations as Semantic Constructs in anObject-Oriented Language.

In OOPSLA '87 Conference Proceedings. ACM Press. 、Rumbaugh, J., et al.. 1991. Object-Oriented Modeling and Design.Prentice Hall.）。例えば、フォルダーオブジェクト（すなわちディレクトリ）は、「包含（containment）」関連性を介してドクメントオブジェクトを保持することができる。関連性内の各オブジェクトは「役割名」によって表現される「役割」を持つ。関連性オブジェクトは、関連性内の各オブジェクトを個々に作用させること無く、関連性によって関連付けられている全てのオブジェクトに影響を及ぼすように作用させることができ、それによって簡潔な表現を用いて複数のオブジェクトを自然な態様で操作することができる。

【００２８】「関連性サービス」は、分散オブジェクトシステムの分散オブジェクト間の関連性の形成と管理とにあたって、分散オブジェクトシステム内の各マシン上で利用可能な設備の集合である。以下の本発明の好ましい実施形態において説明する方法は、ＯＭＧ ＴＣドキュメント94.5.5 に従って関連性サービス内で実行される。しかし、以下に説明する方法と装置は他の仕様に従って設計された関連性サービスを用いて実装実行できること、また、本発明の方法と装置とは単独で、または、関連性サービスの外側で組み合わせて使用できる。

【００２９】ＩＩ．分散オブジェクト環境における同一性チェック

本発明の好ましい実施形態において、分散オブジェクトは、図１の１００に示すような、ネットワークによって相互に連結された１台以上のコンピュータに配置される。図１に示されるように、ネットワーク１００はネットワーク１０４に結合されたコンピュータ１０２を備える。ネットワーク１０４は、更に別のコンピュータ１０８，１１０，および１１２に加えてサーバ、ルータ等１０６を備え、データや命令がネットワーク化されたコンピュータ間で転送できる。こうしたコンピュータネットワークの設計、構成、および実装は当業者には周知である。

【００３０】コンピュータ１０２，１０６，１０８，１１０，および１１２の構成を、図２のコンピュータ２００として示す。各コンピュータは、ランダムアクセスメモリ（ＲＡＭ）２０６に双方向的に、リードオンリメモリ（ＲＯＭ）２０４に単方向的に接続された中央処理装置（ＣＰＵ）２０２を備える。通常、ＲＡＭ２０６は、ＣＰＵ２０２上で現在作動中のプロセスに関する分散したオブジェクトおよびそれらのオブジェクトに関連するデータと命令を含む、プログラミング命令およびデータを格納している。ＲＯＭ２０４は、通常、コンピュータが機能を実行するために用いる基本的動作命令、データ、およびオブジェクトを格納している。更に、ハードディスク、ＣＤ ＲＯＭ、磁気光学（フロプティカル）ドライブ、テープドライブ等の大容量記憶装置２０８が双方向的にＣＰＵ２０２に接続されている。一般には、大容量記憶装置２０８は、アドレス空間がＣＰＵ２０２によって、例えばバーチャルメモリ等としてアクセス可能であるにもかかわらず、ＣＰＵ２０２によって、通常は頻繁に用いられない追加のプログラミング命令、データ、およびオブジェクトを格納する。上記の各コンピュータは、通常、更に、キーボード、ポインタ装置（すなわち、マウスやスタイラス）等の入力媒体を含む入出力源２１０を備える。各コンピュータは、更に、ネットワーク接続２１２を備えることができる。ネットワーク接続２１２を介して、追加の大容量記憶装置（図示せず）をＣＰＵ２０２に接続することができる。上記ハードウェア要素、ソフトウェア要素、およびネットワーキング装置は、当業者にとって周知の標準設計構成である。

【００３１】上記のように分散オブジェクトを用いると、単一のオブジェクトに対して複数のオブジェクト言及が行われたか否かが不確定になる。こうした状態が発生するシナリオを、図３に状態３００として示す。図３に示すように、第１のアドレス空間３０２には、オブジェクトレファレンスＡ３０６を含むプロセス３０４と、オブジェクトレファレンスＢ３０８とが存在する。オブジェクトレファレンスＡは、アドレス空間３１０内に存在するとともに、第１のオブジェクトＯ３１２のアドレスを保有している。オブジェクトレファレンスＢは、アドレス空間３１０内に存在するとともに、第２のオブジェクトＯ’３１４（第１のオブジェクトＯと同一でもよい）のアドレスを保有している。オブジェクトＯとオブジェクトＯ’は異なるアドレス空間にあってもよい。

【００３２】図４において、手順４００は、２つのオブジェクトレファレンス（すなわち図３のオブジェクトレファレンスＡおよびオブジェクトレファレンスＢ）が、夫々、実際に同じオブジェクトを言及しているか否かを、各オブジェクトへユニーク識別子を組み込むことによって区別する方法を示す。イニシャリゼーションステップ４０２から処理を開始し、ステップ４０４で、オブジェクトレファレンスＡによって言及されたオブジェクトがオブジェクトレファレンスＢを受ける。ステップ４０６で、オブジェクトレファレンスＢによって言及された第２のオブジェクトに、オブジェクトレファレンスＡによって言及された第１のオブジェクトによって、第２のオブジェクトのユニーク識別子（$ＩＤ_{Ｂ}$）の問合せがなされる。次に、ステップ４０８で、第１のオブジェクトは、自らのユニーク識別子（$ＩＤ_{Ａ}$）が第２のオブジェクトの識別子のユニーク識別子と同一であるか否かを判定する。もし、２つの識別子が等しければ、肯定の応答がステップ４１０で返される。等しくなければ、否定の応答がステップ４１２で返される。手順４００はステップ４１４で終了する。

【００３３】ユニーク識別子を有するオブジェクトの作成を図５に示す手順５００で説明する。イニシャリゼーションステップ５０２から開始し、ステップ５０４にお

けるオブジェクトのユニークＩＤの作成は、ステップ５０６におけるオブジェクト自身の作成に先行する。ステップ５０８で、ユニークＩＤはそのオブジェクトに関して適当な場所に格納され、ステップ５１０で、完成されたオブジェクトはユーザーに返され、その後、手順はステップ５１２で終了する。ユニークＩＤ作成のステップとオブジェクト作成のステップとは逆にしてもよい。言い換えれば、ユニークＩＤを作成する前に、オブジェクトを作成することができる。オブジェクト識別子はオブジェクトのリファレンスデータエクステンション（reference data extension）またはオブジェクトに関連する不変データ格納装置（persistentdata strage）に格納されることが好ましい。なお、ユニークＩＤはオブジェクトへのアクセスが可能な任意の場所に保存すればよい。

【００３４】ＩＤの作成を、図６に示す手順６００で説明する。図６に示すように、イニシャリゼーションステップ６０２に続いて、ステップ６０４で、オブジェクトがその上で作成されるマシンに関するネットワークアドレスが得られる。次に、ステップ６０６で、識別可能なオブジェクトを作成しているプロセスに関するプロセスＩＤが得られ、ステップ６０８で、現在のローカルシステム時間が得られる。更に、ステップ６１０で、このプロセスに関連するインクリメント器（incrementor）からの値が得られ、マシンアドレス、ステップ６１２で、プロセスＩＤ、ローカルシステム時間、およびインクリメント値が連鎖させられてシングルストリングが形成され、このストリングは、上述のステップ５０８と同様に、オブジェクトへのアクセスが可能な記憶場所に格納される。好ましい記憶場所としては、オブジェクトに関する不変ストーレッジのために割り当てられたメモリ空間のような、効率的にアクセスができる場所があげられる。不変データ格納装置を用いることは、当業者には周知であろう。手順６００の実行は、ステップ６１４で終了する。

【００３５】好適なマシンアドレスはそのマシンに関するネットワークＩＰ（インターネットプロトコル）アドレスであり、当業者にとっては周知であろう。プロセスＩＤはオブジェクトは、そのオブジェクトが形成されているマシンのオペレーティングシステムに問合せれば得られる。現在のシステム時間は、通常、システムのオペレーティングシステムへのコールによって得られる。インクリメント値６１０は、同時に実行されている種々のスレッド（threads）を区分する（differentiate）ローカルカウンタへのコールによって得られる。これらの手順はマルチスレッドコンピューティングに精通した当業者には周知であろう。

【００３６】上記の手順は、ＯＭＧ　ＴＣドキュメント94.5.5「関連性サービス仕様」に記した仕様に従って好適に実装実行される。好適な関連性サービスは下記のように定義される CosObjectIdentity モジュールをサポートする：

```
module CosObjectIdentity{
    typedef string ObjectIdentifier;
    interface IdentifiableObject {
        readonly attribute ObjectIdentifier
constant random id;
         boolean is identical (
             in IdentifiableObject other object);
        };
    };
```

IdentifieableObjectインターフェイスをサポートするオブジェクトは、タイプObjectIdentifierの属性とis identical操作を実行する。ObjectIdentifi-erは、図５と図６とに示した上述の手順に従って、各オブジェクトに関して定義される。各マシンは、予め定義されたネットワークアドレスを有するので、マシンアドレスはネットワーク上の全てのマシンについて、１つのユニーク識別子を提供する。プロセスＩＤは、同じマシン上で実行中の全プロセスにわたってユニークに提供される。しかし、ＵＮＩＸマシン上のプロセスＩＤは、時間が経過するとユニークでなくなるかも知れず、したがって、縮退の可能性のあることは当業者にとって周知である。この縮退は、種々の時刻に開始されるプロセスに関するローカルシステム時間を参照することによって是正される。しかし、複数のストリングが同じプロセス上でほぼ同時※に開始される場合、すなわち開始時刻がシステムクロックの精度（granularity）より小さい値だけ異なる場合、インクリメント器は時間に依存しない各スレッドに関する識別子を提供する。したがって、上記全ての情報を担う１つのストリングが、分散オブジェクト環境における他のオブジェクトに関してユニークな１つのオブジェクトのための識別子を提供するものと見られる。

【００３７】他のオブジェクトの識別子をそれ自身の識別子と比較するため、問合せを発行するオブジェクトは、図４のステップ４０８において is identical 操作を行うことが好ましい。このis identical操作は、次のように定義される：

```
boollean is_identical (
    in IdentifiableObject other_object);
```

この操作は、オブジェクトが他のオブジェクト（other_

object）と同一であれば、すなわち両オブジェクトが同じ識別子を持つならば「正」を返す。同一でなければ、この操作は「誤」を返す。

【００３８】ＩＩＩ．オブジェクト間の関連性形成に対する無矛盾性チェック

本発明は、複数のオブジェクト間に形成するべき関連性の論理的無矛盾性をチェックするための方法と装置を含む。

【００３９】図7は、3つの異なるオブジェクトの関連性を示す。図7（ａ）の関連性７００に示すように、関連性は、ドキュメントオブジェクト７０２、グラフィックオブジェクト７０４、およびフォルダオブジェクト７０６の間に示されている。ドキュメントオブジェクト７０２とグラフィックオブジェクト７０４とは関連性オブジェクト７０８を介して関連付けられている。フォルダーオブジェクト７０６とドキュメントオブジェクト７０２は関連性オブジェクト７１０を介して関連付けられている。また、各オブジェクトは、当該オブジェクトが関与する関連性の各タイプに関する関連する役割を有する。したがって、ドキュメントオブジェクト７０２は関連性７０８に関して関連する役割７１４を有するとともに、関連性７１０に関して関連する役割７１８を有する。フォルダオブジェクト７０６は関連性７１０に関して役割７１６を有し、グラフィックオブジェクト７０４は関連性７０８に関して関連する役割７１２を有する。関連性７０８と７１０は、通常、「包含」関連性と呼ばれる。役割７１２と役割７１８とは、通常、「包含された」役割と呼ばれ、役割７１６と役割７１４とはしばしば「含む」役割と呼ばれる。オブジェクト間に関連を形成する関連性オブジェクトを用いることは公知である（Rumbaugh, J. 1987. Relations as Semantic Constructs in an Object-Oriented Language. In OOPSLA '87 Conference Proceedings. ACM Press.；Rumbaugh, J., et al.. 1991. Object-Oriented Modeling and Design. Prentice Hall.；Martin, B.E., and Cattell, R.G.G. 1994. Relating Distributed Objects. In Proceedings of the 20th VLDB Conference, Santiago, Chile, 1994.参照）。

【００４０】関連性オブジェクトを使うことにより、オブジェクトを修正すること無く、各オブジェクトを種々の役割において種々の関連性を介して関連付けることができるので、オブジェクト７０２、７０４、および７０６の各々の最も一般化された使用を可能にすることできる。新しい役割は、各新しい関連性に関してオブジェクトと簡単に関連付けられる。したがって、例えば、図7（ｂ）に示すように、フォルダー、ドキュメント、およびグラフィックの各オブジェクトの代替関連性である関連性７３０とできる。ここで、フォルダーオブジェクト７３２は、包含関連性オブジェクト７３８と包含関連性オブジェクト７４０とを夫々介して、ドキュメントオブジェクト７３４とグラフィックオブジェクト７３６とを保有する。この例においては、フォルダオブジェクト７３２は唯一の役割、すなわち、「含む」役割を有し、ドキュメントオブジェクト７３４とグラフィックオブジェクト７３６とは、夫々「含まれる」役割を有する。

【００４１】しかし、論理的無矛盾性を保つため、いくつかの関連性の形成を防止しなければならない。例えば、あるソフトウェアシステムは、フォルダがドキュメントに含まれる関連性を防止する必要があるかも知れない。更に、形成された各関連性が所要数の役割を確実に含んでいる必要がある。例えば、ある包含関連性は唯一の「含む」関連性を持って形成されないかも知れない。最後に、関連性の濃度（cardinality）、すなわち、ある役割に結合された関連性の数は無矛盾でなければならない。その理由は、例えば、単一のドキュメントが複数回のネストができない（un-nested）複数のフォルダを含む関連性を許すことは論理的無矛盾性を破ることになるからである。したがって、数、タイプ、および濃度に関して無矛盾な関連性のみが形成される。

【００４２】周知のように、関連性オブジェクトはファクトリオブジェクトを使用することにより形成され、このファクトリオブジェクトは、関連付けられるべきオブジェクトとそれらの各役割をアーギュメントとして受け、適当な関連性オブジェクトを出力する。こうしうた関連性ファクトリの形成を、図8に示す手順８００で説明する。イニシャリゼーションステップ８０２から開始し、ステップ８０４で、関連性ファクトリオブジェクトを介して関連付けられるべきオブジェクトに関する役割タイプが、関連性ファクトリを形成するファクトリオブジェクトによって受け取られる。次に、ステップ８０６で、関連性ファクトリオブジェクトが作成され、ステップ８０８で、役割タイプが関連性ファクトリオブジェクト内に期待アレイ［$E_1$，．．．，$E_N$］として格納される。ここで、$E_i$はｉ番目に関して保存された役割タイプであり、Ｎは関連性の自由度である。その後、プロセスはステップ８１０で終了する。

【００４３】次に、図9の手順９００を参照して、形成されるべき関連性の論理的無矛盾性のチェック方法を説明する。イニシャリゼーションステップ９０２から開始し、ステップ９０４で、関連付けられるべきオブジェクトの役割が、関連性ファクトリオブジェクトによって受け取られる。ステップ９０６で、形成されるべき関連性に関して適切な数の役割が受け取られたか否かの判定が行われる。適切な数の役割が受け取られていない場合、すなわち、エラーが発生した場合、ステップ９０８で、エラーが報告され、この手順はステップ９１０で終了する。

【００４４】一方、ステップ９０６で、適切な数の役割が受け取られた場合、次に、ステップ９１２で、各役割に関するタイプチェックが行われ、ステップ９１４で、

タイプチェックが期待される出力に一致するか否かの判定が行われる。関連付けられるべきオブジェクトに関する役割タイプは、オブジェクトのインターフェイスの記述とそれらのタイプＩＤとを保持しているインターフェイス保存場所から検索されることが好適である。代替として、narrow() 操作または is a() 操作を用いてオブジェクトに直接問合せることができる。期待に一致しない場合、ステップ９０８でエラーが返され、処理はステップ９１０で終了する。一方、タイプチェックが期待に一致した場合、次に、ステップ９１６で、関連性オブジェクトが形成される。ステップ９１８で、関連性オブジェクト形成ステップ中で濃度エラーが返されたか否かのチェックが行われ、チェック結果が正しければ、ステップ９０８において適当なエラーが返される。濃度エラーがなければ、次に、形成された関連性オブジェクトは９２０で返される。

【００４５】タイプチェックステップ９０８を、図１０に示す手順１０００で、更に詳細に説明する。イニシャリゼーションステップ１００２から開始し、役割タイプの数と等しいディメンションを有するタイプチェックアレイをイニシャライズする。すなわち、ステップ１００４で、全てのアレイコンポーネントをゼロにセットする。ステップ１００６で、期待アレイ［$E_1$，．．．，$E_N$］が読み出され、第１のループが定義され、このループのループインデックスはファクトリオブジェクトによって受け取られる役割の数Ｎの全部に及ぶ。第２のループ１００８は読み出された役割タイプの全数にわたり、そのインデックスは ｊ で定義される。ここで 、ｊ＝１～Ｎである。サブループ１００８内におけるステップ１０１０で、受け取られた役割タイプ（$R_i$）が読み出された役割タイプ（$E_j$）と同一であるか否かの判定が、機能 Ri→is_a($E_j$) をコールすることによって行われる。上記のように、役割タイプはインターフェイス保存場所から、あるいはオブジェクトから直接得られる。ステップ１０１０での判定が正の場合、タイプチェックアレイのｊ番目の要素に含まれている値は、ステップ１０１２においてインクリメントされる。一方、正でない場合には、ループインデックスｊがインクリメントされる。次に受け取られた役割タイプが次に格納された役割タイプに対して試験される。Ｎ個全ての格納された役割タイプが調べられると、ループインデックスｉがインクリメントされ、次に受け取られた役割タイプが全ての保存された役割タイプに対してチェックされる。全ての受け取られた役割タイプが全ての保存された役割タイプに対してチェックされると、すなわち、両ループインデックス（ｉおよびＪ）が１からＮまで実行されると、プロセスはステップ１０１４で終了する。

【００４６】関連性オブジェクトの作成、すなわち、図９のステップ９１６を図１１の手順１１００で更に詳細に説明する。ステップ１１０２から開始し、ステップ１１０４で、所望の関連性オブジェクトが形成され、ステップ１１０６で、関連性オブジェクトにリンクされるべきＭ個の役割にわたるループが開始する。ループ１１０６内において、ステップ１１０８で、ｉ番目の役割が$Role_i$→Link(Relationship)操作によって関連性オブジェクトにリンクされる。ステップ１１１０で、$Role_i$→Link(Relationship) 操作が濃度エラーを返したか否かの判定が行われる。エラーが返されなかった場合、ステップ１１１２で、リンクされた関連性は保存され、ステップ１１０４においてループがインクリメントすると次の役割が読み出される。Ｍ個全ての役割が問題なくリンクさせられると、ステップ１１２０で、完成された関連性オブジェクトは返される。

【００４７】一方、ステップ１１１０で濃度エラーが返された場合、ステップ１１１４で、関連性オブジェクトが破棄され、ステップ１１１８で手順が終了する前に、ステップ１１１６で、濃度エラーが返される。上記の、関連性オブジェクトを形成するステップ、オブジェクトへ役割をリンクするステップ、および、数、タイプ、および濃度の制約をチェックするステップの各ステップが公知の技法を用いて実装できることは、当業者であれば理解可能であろう。

【００４８】したがって、例えば、図７（ａ）の説明にあたって述べた関連性においては、包含関連性７０８または包含関連性７１０のいずれかに関し、関連性ファクトリに保存された期待される役割タイプのシーケンスは、{$E_1$, $E_2$}=(CONTAINS, CONTAINED IN)であろう。これは、一方の役割タイプは、含む（contains）役割であり、他方は、含まれる（contain-ed in）役割でなければならない。受け取られた役割タイプも{$R_1$,$R_2$}={CONTAINS,CONTAINED IN}であろう。タイプチェックアレイは 1x2 アレイで、一方のコラムが関連性ファクトリによって受け取られた含む役割の個数を表すとともに、他方のコラムが受け取られた含まれる役割の個数を表す。最初は、タイプチェックアレイは [0,0]であろう。これは、含む役割も含まれる役割も識別されなかったことを意味する。期待アレイも 1x2 アレイであろうが、その値は[1,1]であって、含む役割または含まれる役割が各１個づつ期待されることを表す。

【００４９】引き続き、この例において、関連性ファクトリによって受け取られた第１の役割($R_1$)が含む役割である場合、ダブルループ（i=j=1）を通る第１パス上において、$R_1$ と $E_1$ とは共に含む役割であるので、コール $R_1$→is a($E_1$)は値１すなわち「正」を返すであろう。次に、タイプチェックアレイの第１要素はインクリメントされて［1,0］を発生する。しかし、第２パス（i=1,j=2）においては、コール $R_1$→is a($E_2$)はゼロすなわち「誤」を返すであろう。そして、タイプチェックアレイは変化しないであろう。

【００５０】同様に、ループインデックス i,i=2 を通

る第2パスにおいては、受け取られた第2の役割（$R_2$）は含まれる役割であるので、コール $R_2$→is a($E_1$)はゼロすなわち「誤」を返すであろう。そして、タイプチェックアレイは変化しないであろう。しかし、最終パスにおいては、$R_2$と$E_2$とは共に含まれる役割であるので、コール$R_2$→is a($E_2$)は値1すなわち「正」を返すであろう。このようにして、タイプチェックアレイの第2要素はインクリメントされて最終結果〔1,1〕を発生する。再び、図9のステップ９１０を参照して、タイプチェックアレイを期待アレイに対して比較すると「正」値が発生し、関連性オブジェクトが形成されるであろう。

【００５１】しかし$R_1$と$R_2$のいずれもが含む役割である場合、j=1のときのみ正の結果が返されるので、タイプチェックアレイは最終値〔2,0〕を有するであろう。この条件下で、期待アレイとタイプチェックアレイの比較は誤を発生し、エラーが返されるであろう。同様に、$R_1$または$R_2$が含むでも含まれるでもない場合、やはりエラーが返されるであろう。



＊【００５２】いったん、タイプと数のチェックが問題なく完了すると、関連性オブジェクトが形成され、役割は上記のように関連オブジェクトにリンクされる。この時点で、関連性オブジェクトにリンクされるべき役割の各々について、最小と最大の濃度制約が確実に充たされるように、濃度がチェックされる。ここでは、最小濃度とは、ある役割が関与しなければならない関連性の事例の最小数である。最大濃度とは、ある役割が関与することのできる関連性の事例の最大数である。この例においては、役割$R_1$と役割$R_2$とは、夫々、少なくとも1つの関連性に関与しなければならない。

【００５３】上記の例で概説したステップは、ＯＭＧＴＣドキュメント94.5.5 に従って、CosRelatioshipsモジュールのRelationshipFactoryンターフェイスを用いて実装実行されることが好適である。このインターフェイスは、形成されるべき関連性における関連付けられたオブジェクトを表す指名された一連の役割をアーギュメントとする、下記の作成操作を定義する：

```
Relationship create (in NamedRoles named roles)
    raises (RoleTypeError,
        MaxCardinalityExceeded,
        DegreeError,
        DuplicationRoleName,
        UnknownRoleName);
```

CosRelationships モジュールの形成操作を用いて、関連性において関連付けられたオブジェクトを表す指名された役割を転送する。この例においては、それら役割は、フォルダ、ドキュメント、および／またはグラフィックオブジェクトに関連する含む役割と含まれる役割とである。関連性ファクトリへ転送された役割の数が過多または過少であったかの判定が、関連性ファクトリから DegreeErrorexception を発生させる。関連性ファクトリへ転送された役割タイプが誤りであるとの判定により、RoleTypeError exception が発生する。濃度が過剰であるとの判定により MaxCardinality-Exceeded error が発生する。ミニマム濃度はchack minimum cardinality() 操作を用いてチェックすることができる。役割の数、役割タイプ、および役割の濃度が関連性と一致していた場合、ファクトリは、関連性において関連付けられているオブジェクトを表す関連性と１セットの役割とを役割へ転送するリンク操作とを用いて関連性オブジェクトを形成し、かつ役割を通報する。

【００５４】ＩＶ．オブジェクトおよび役割に関しオブジェクトレファレンスを関連性において格納保存すること

本発明は、特定のオブジェクトの役割における関連性によって関連付けられているその特定のオブジェクトに関する役割とオブジェクトレファレンスを格納する方法と装置とを含む。この格納によっては、オブジェクトレファレンスと関連性における他の各オブジェクトへの役割※が任意のひとつの役割の中に見いだされるので、関連性を介してのナビゲーションの効率を高める。こうした格納がなければ、役割は他のオブジェクトの各々へリクエストを送らねばならず、関連性の往復に係わるシステムのオーバーヘッドが増加する。

【００５５】図１２の手順１２００で上記を説明する。手順１２００は、イニシャリゼーションステップ１２０２から開始し、ステップ１２０４で、関連性オブジェクトRへのオブジェクトレファレンスと、関連性オブジェクトRによって第2のオブジェクト役割を有する第2のオブジェクトへ関連付けられている第1のオブジェクトの役割名ＲＮとが、第2のオブジェクト役割に受け入れられる。ステップ１２０６で、第1のオブジェクトへのオブジェクトレファレンスが、ルックアップテーブル（ＬＵＴ）または第2の役割に関連する保存場所に見いだされるか否かを判定する。答が正ならば、ステップ１２０８で、オブジェクトレファレンスが、第1のオブジェクトのために返され、プログラム機能はステップ１２１０において終了する。誤であれば、ステップ１２１２で、関連性は役割名ＲＮを問い合され、次に、ステップ１２１４で、役割ＲＮは第1のオブジェクトのオブジェクトレファレンスを問い合される。ステップ１２０６で、このオブジェクトレファレンスが言及したルックアップテーブルに保存される。

【００５６】上記の方法は、以下に示す形式を有するコール get other related object()を用いて、上記の Co

sRelationships モジュールの Role Interface（役割インターフェイス）に従って実装実行されることが好適である。

【0057】

```
RelatedObject get_other_related_object (
        in RealtionshipHandle rel,
        in RoleName target_name)
    raises (UnknownRoleName,
        UnknownRelationship);
```

この操作は、target name to the related object と名付けられた役割によって表される関連付けられたオブジェクトへの関連性 rel をナビゲートする。問い合された役割の保存場所に target name がない場合、例外 UnknownRoleName が取り上げられる。役割が関連性を認識しない場合、UnknownRelationship が取り上げられる。

【0058】一例として、図7（a）で説明したフォルダードキュメント関連性を用いた場合には、コール get other related object() をフォルダーオブジェクト706の含む役割716へ転送して、アーギュメントとしてオブジェクトレファレンス対包含関連性710と役割名「含む（contains in）」を用いて、役割オブジェクト716に関連するルックアップテーブル内のドキュメント702に関するオブジェクトレファレンスを調べるように役割716を促す。ドキュメントオブジェクト702に関するオブジェクトレファレンスが見つかれば、オブジェクトレファレンスが返される。もし、オブジェクトレファレンスが記録されなければ、関連性の役割を返す named roles attribute を用いて包含関連性710が問い合わされる。次に、この役割は、関連性オブジェクトのオブジェクトレファレンスを判定するため、related object attribute を用いて問い合わされる。このオブジェクトレファレンスは、ルックアップテーブルに保存され返される。

【0059】第1の役割に1つ関連性において関連付けられた1つの役割に関するオブジェクトレファレンスは、類似のやり方で判定される。命令の実行手順は、ステップ図13に示す手順1300と同様の手順である。手順300は、ステップ1302から開始する。ステップ1304で、関連性オブジェクトRへのオブジェクトレファレンスと役割名RNはステップ1304における第1の役割へ転送される。ステップ1306で、役割名RNに関してRによって関連付けられている役割が、既にルックテーブルに保存されたか否かの判定が行われる。答が「正（yes）」の場合、ステップ1308で、当該役割に関するオブジェクトレファレンスが返され、ステップ1310で、判定が行われる。答が「誤（no）」の場合、ステップ1312で、関連性は役割名RNに関して問合せられ、当該役割に関するオブジェクトレファレンスが検索され、ステップ1314で、ルックアップテーブルに格納される。



【0060】上記の方法は、以下に示す形式を有するコール get other role() を用いて、上記の CosRelationships モジュールの役割インターフェイスに従って実装実行されることが好適である。

【0061】

```
Role_get other_role (
        in RealtionshipHandle rel,
        in RoleName target_name)
    raises (UnknownRoleName,
        UnknownRelationship);
```

この操作は、target name to the role と名付けられた役割によって表される関連付けられたオブジェクトへの関連性 rel をナビゲートする。問合せられた役割の保存場所に target name がない場合、例外 UnknownRoleName が取り上げられる。役割が関連性を認識しない場合、UnknownRelationship が取り上げられる。

【0062】上記の方法は、上記 CosRelationships モジュールの役割インターフェイスに従って実装実行されることが好適である。一例として、図7（a）において説明したフォルダードキュメント関連性を用いて説明する。コール get other role() をフォルダーオブジェクト706の含む役割716へ転送し、アーギュメントとしてオブジェクトレファレンス対包含関連性710と役割名「contains in」を用いて、役割オブジェクト716とともに格納されたルックアップテーブル内のドキュメント702に関する オブジェクトレファレンス を調べるように役割716を促す。contains in 役割オブジェクト718に関するオブジェクトレファレンス が見つかれば、 オブジェクトレファレンス が返される。もし、オブジェクトレファレンス が記録されなければ、当該関連性の役割を返す named roles attribute を用いて包含関連性710が問合せられる。このオブジェクトレファレンスはルックアップテーブルに格納されて返される。

【0063】以上、詳細に説明したように、本発明は、分散オブジェクトシステム全体にわたるオブジェクト間の関連性を管理するための重要な方法を提供する。本発明の方法を用いれば、以下のことが可能である。すなわち、ネットワーク全体にわたる種々のメモリ位置に格納されている分散したオブジェクトがユニークに識別されること、関連性がオブジェクト間で無矛盾に定義されること、およびプロセス操作が、関連するオブジェクトの役割がその役割に関連付けられたオブジェクトに向けられているオブジェクトレファレンスを含む自動格納を用いることによって速められることが可能である。

【図面の簡単な説明】

【図1】相互接続された種々のコンピュータを備えるコンピュータネットワークシステムの構成図である。

【図2】図1におけるコンピュータの構成図である。

【図3】コンピュータで実装実行されるプロセスに関し

て、遠隔のメモリ位置にあるオブジェクトを指し示す2つのオブジェクトレファレンスがある状態の説明図である。

【図4】2つのオブジェクトレファレンスが同一のオブジェクトを指し示しているか否かを判定する方法手順のフローチャートである。

【図5】1つのオブジェクトに関してユニーク識別子を有するオブジェクトを形成する手順のフローチャートである。

【図6】図5のステップ504の詳細フローチャートである。

【図7】オブジェクト間の関連性の説明図である。

【図8】関連性オブジェクトを形成するファクトリオブジェクトにおける格納された役割タイプを提供する手順のフローチャートである。

【図9】妥当な数の役割と役割タイプが複数のオブジェクト間に関連性を形成するのに有効なファクトリオブジェクトへ転送されたか否かを判定するための方法手順のフローチャートである。

【図10】図9のステップ908の詳細フローチャートである。

【図11】図9のステップ916の詳細フローチャートである。

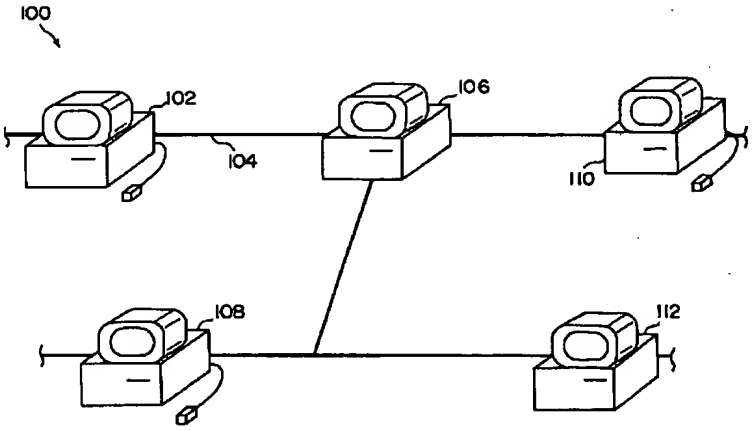
【図12】オブジェクトレファレンスを、1つの関連性について、1つのオブジェクトに格納する手順のフローチャートである。

【図13】1つのオブジェクトレファレンスを、ある関連性について、1つのオブジェクトに関連付けられている役割に格納する手順のフローチャートである。

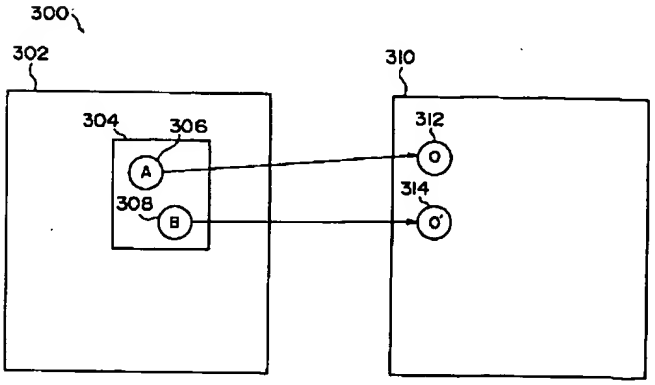
【符号の説明】

100…ネットワーク、102，106，108，110，112…コンピュータ、104…ネットワーク接続、200…コンピュータ、202…中央処理装置（CPU）、204…ROM、206…RAM、208…大容量記憶装置、210…入出力源、212…ネットワーク接続、302…アドレス空間、304…プロセス、306，308…オブジェクトレファレンス、310…アドレス空間、312，314…オブジェクト。

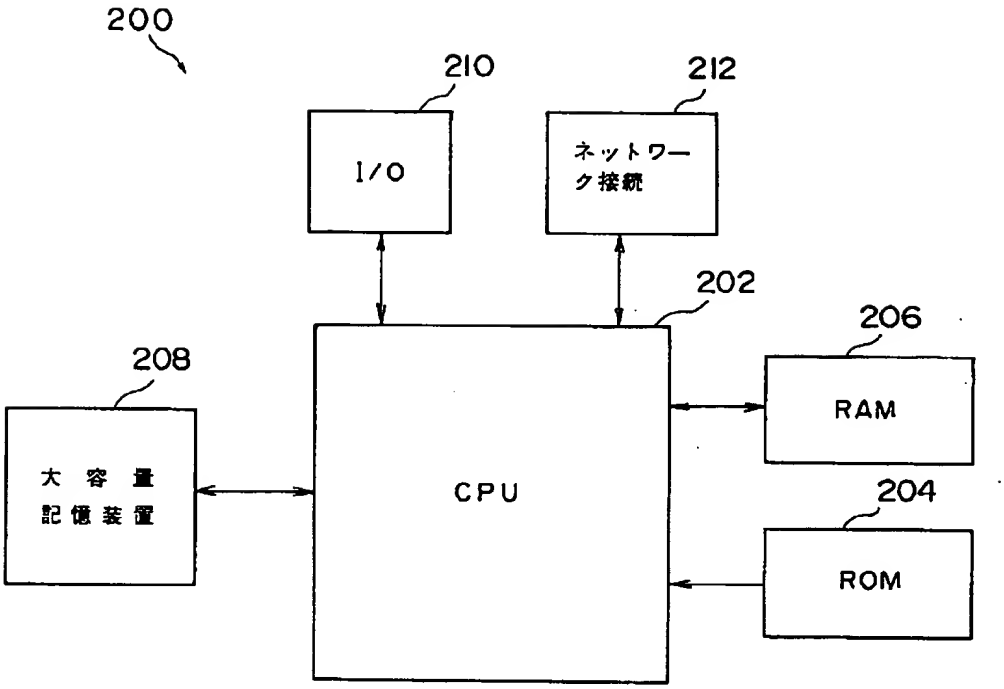
【図1】



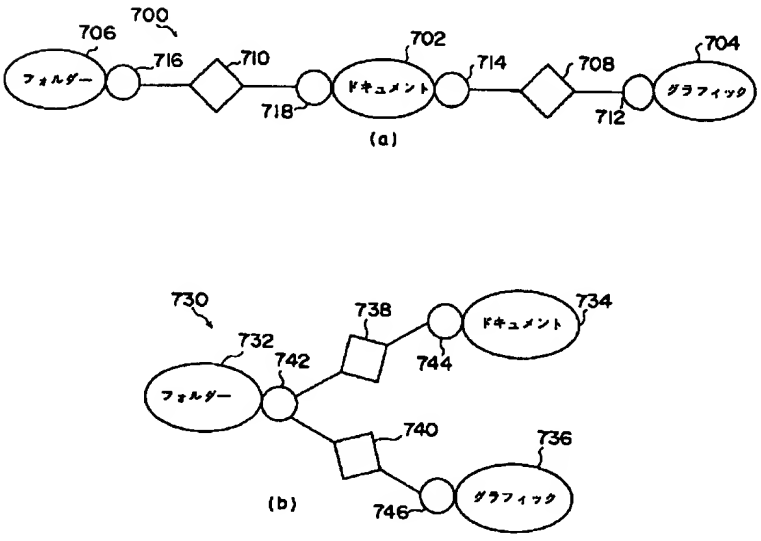

【図3】



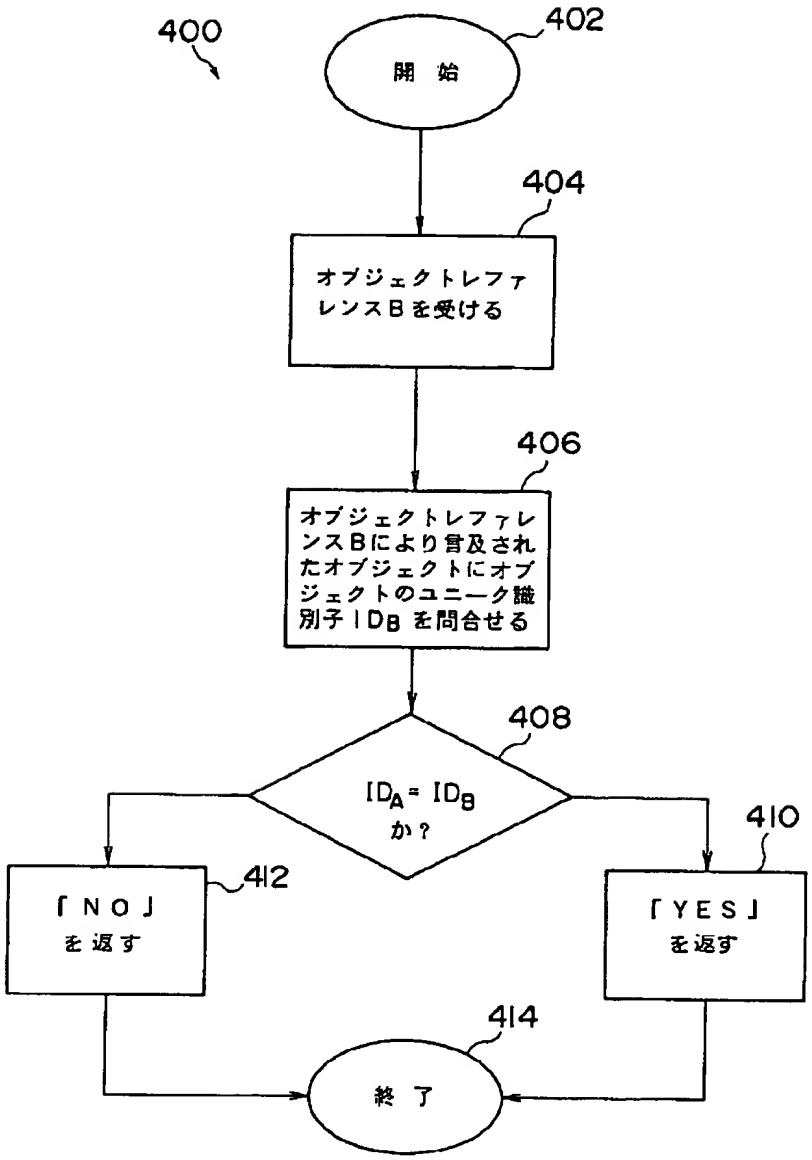
【図2】


200
210
I/O
212
ネットワーク接続
202
206
RAM
208
大容量記憶装置
CPU
204
ROM

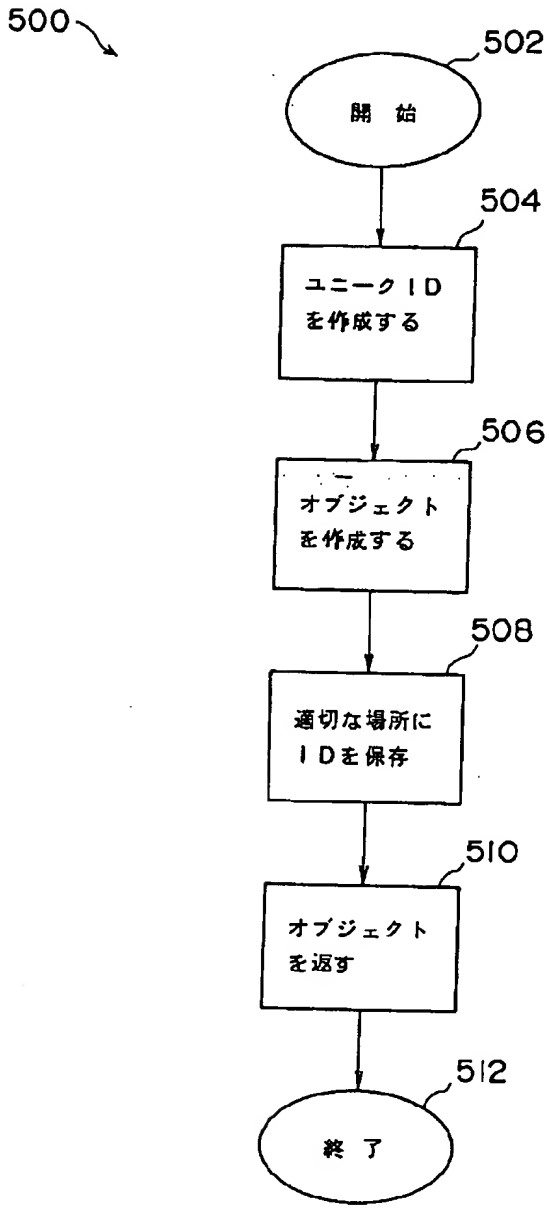
【図7】


700
706
フォルダー
716
710
718
702
ドキュメント
714
708
712
704
グラフィック
(a)
730
732
フォルダー
742
738
744
ドキュメント
734
740
746
736
グラフィック
(b)

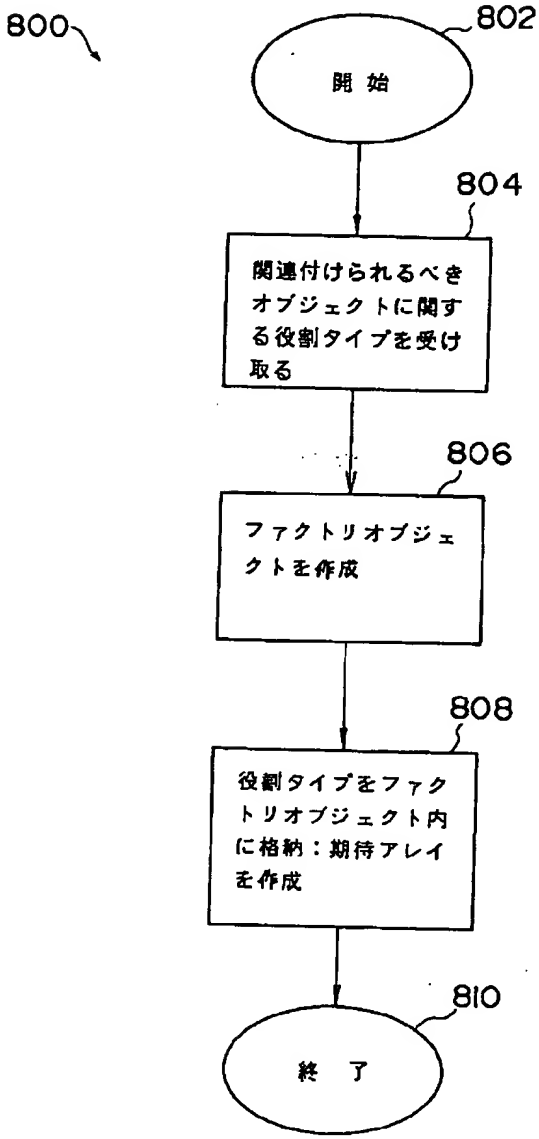
【図4】


400
402
開　始
404
オブジェクトレファレンスBを受ける
406
オブジェクトレファレンスBにより言及されたオブジェクトにオブジェクトのユニーク識別子 $ID_B$ を問合せる
408
$ID_A = ID_B$ か？
410
「YES」を返す
412
「NO」を返す
414
終　了

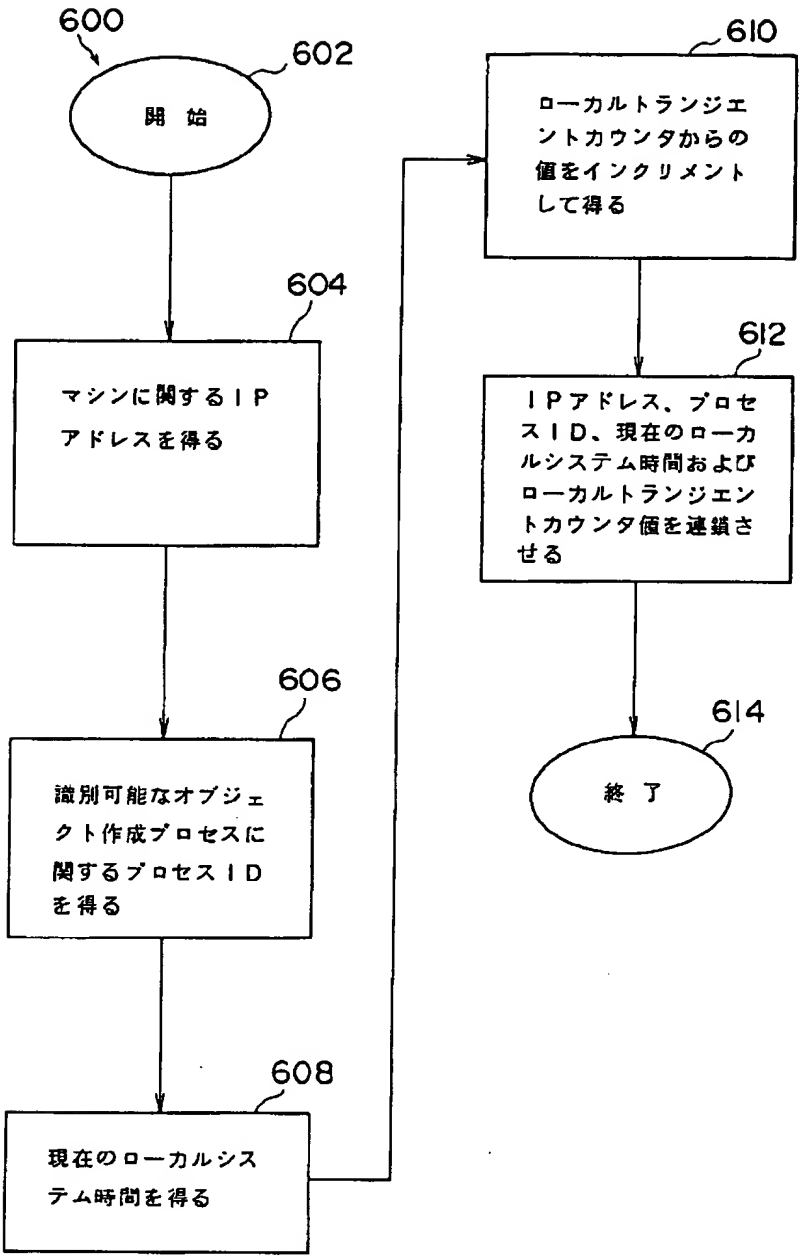
【図5】


500
502
開 始
504
ユニークＩＤを作成する
506
オブジェクトを作成する
508
適切な場所にＩＤを保存
510
オブジェクトを返す
512
終 了

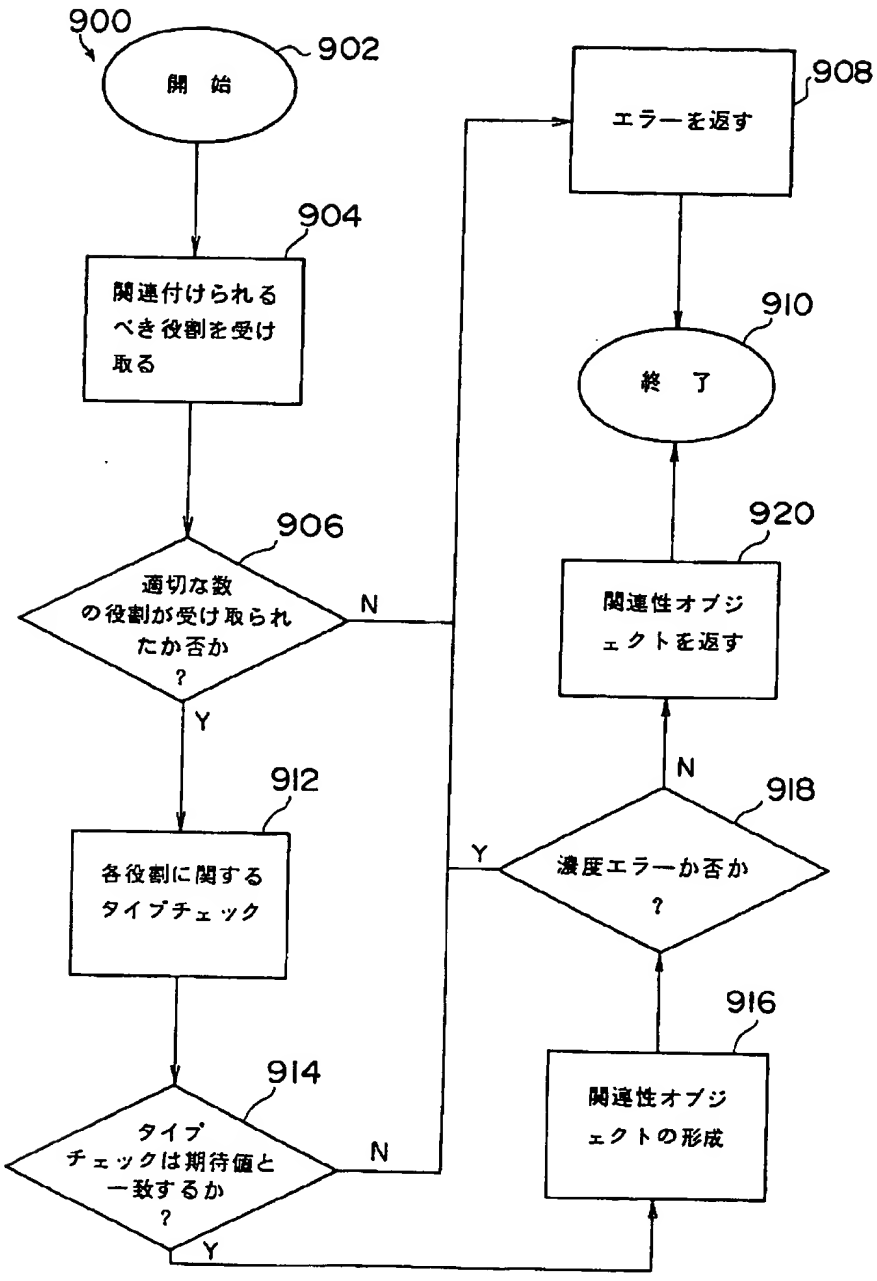
【図8】


800
802
開 始
804
関連付けられるべきオブジェクトに関する役割タイプを受け取る
806
ファクトリオブジェクトを作成
808
役割タイプをファクトリオブジェクト内に格納：期待アレイを作成
810
終 了

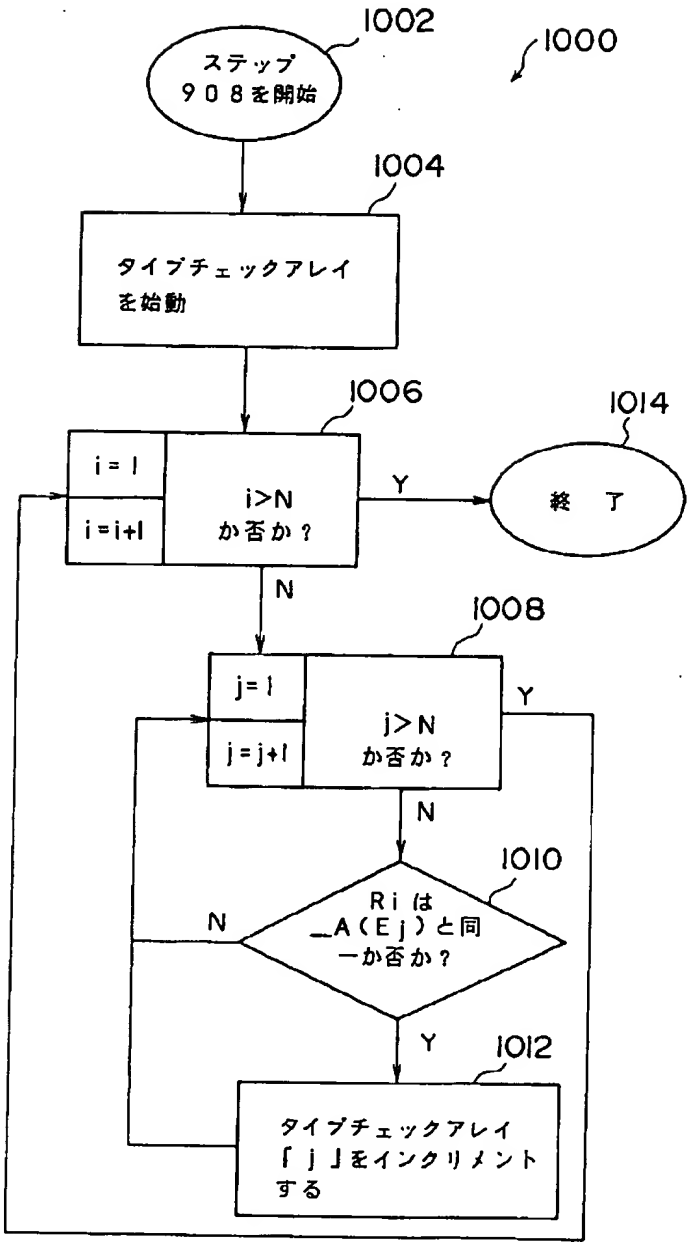
【図6】


600
602
開　始
604
マシンに関するＩＰアドレスを得る
606
識別可能なオブジェクト作成プロセスに関するプロセスＩＤを得る
608
現在のローカルシステム時間を得る
610
ローカルトランジエントカウンタからの値をインクリメントして得る
612
ＩＰアドレス、プロセスＩＤ、現在のローカルシステム時間およびローカルトランジエントカウンタ値を連鎖させる
614
終　了

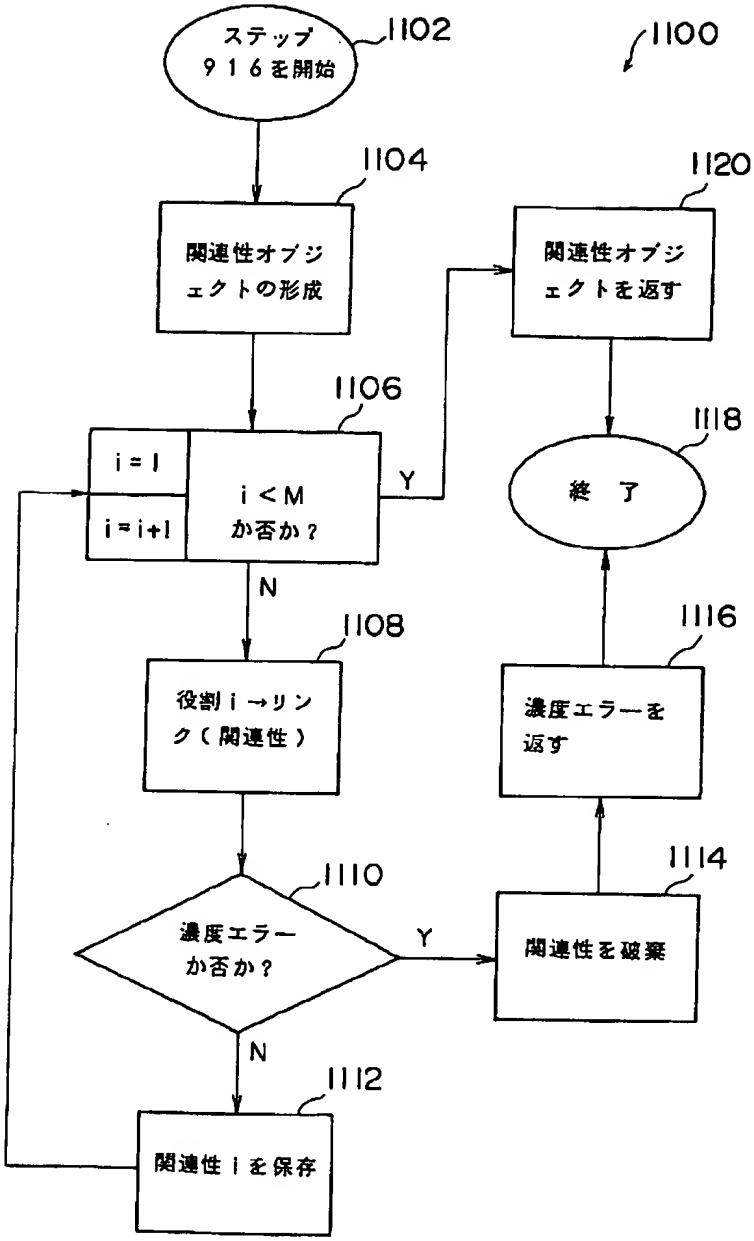
【図9】


900
902
開　始
904
関連付けられるべき役割を受け取る
906
適切な数の役割が受け取られたか否か？
N
Y
912
各役割に関するタイプチェック
914
タイプチェックは期待値と一致するか？
N
Y
908
エラーを返す
910
終　了
920
関連性オブジェクトを返す
N
918
濃度エラーか否か？
Y
916
関連性オブジェクトの形成

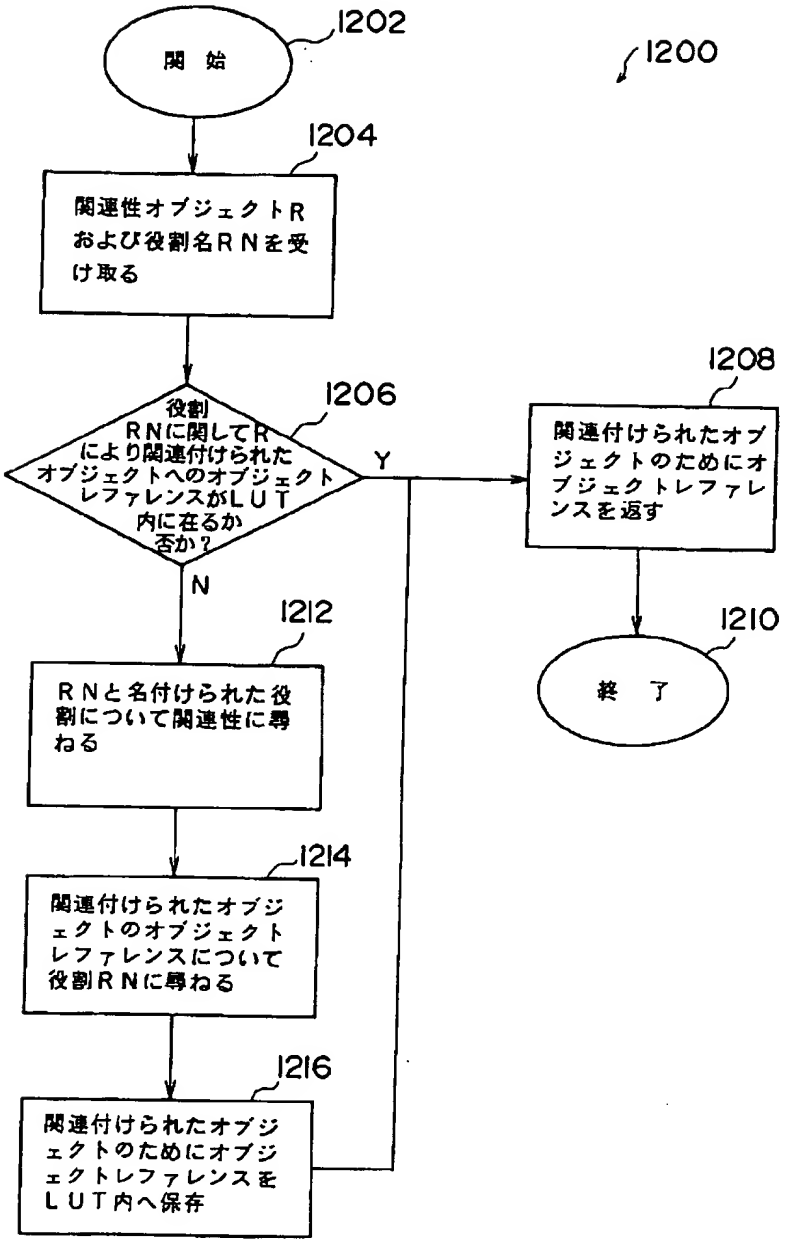
【図10】


1002
1000
ステップ
９０８を開始
1004
タイプチェックアレイ
を始動
1006
1014
i = 1
i=i+1
i>N
か否か？
Y
終　了
N
1008
j= 1
j=j+1
j>N
か否か？
Y
N
1010
Ri は
_A(Ej)と同
一か否か？
N
Y
1012
タイプチェックアレイ
「ｊ」をインクリメント
する

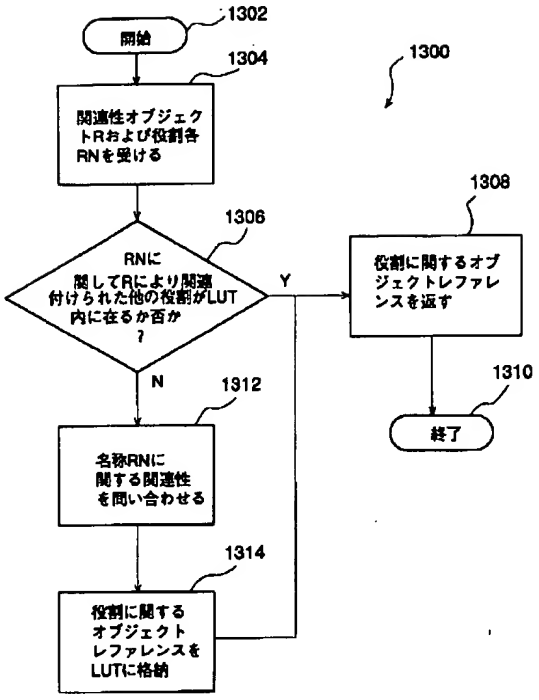
【図11】


ステップ
916を開始
1102
1100
1104
関連性オブジェクトの形成
1120
関連性オブジェクトを返す
1106
i = 1
i = i+1
i < M
か否か？
Y
N
1118
終　了
1108
役割 i →リンク（関連性）
1116
濃度エラーを返す
1110
濃度エラーか否か？
Y
1114
関連性を破棄
N
1112
関連性 i を保存

【図12】


1200
1202
開　始
1204
関連性オブジェクトRおよび役割名RNを受け取る
1206
役割RNに関してRにより関連付けられたオブジェクトへのオブジェクトレファレンスがLUT内に在るか否か？
Y
N
1208
関連付けられたオブジェクトのためにオブジェクトレファレンスを返す
1210
終　了
1212
RNと名付けられた役割について関連性に尋ねる
1214
関連付けられたオブジェクトのオブジェクトレファレンスについて役割RNに尋ねる
1216
関連付けられたオブジェクトのためにオブジェクトレファレンスをLUT内へ保存

【図13】




フロントページの続き

(72)発明者　ジェファーソン　エー．　ディンキンス
アメリカ合衆国，　カリフォルニア州　94087，　サニーヴェール，　エヌ．　サージェン　1355

(72)発明者　マーク　ダブリュー．　ハプナー
アメリカ合衆国，　カリフォルニア州　95125，　サン　ノゼ，　ブルックス　アヴェニュー　595